



Title	データベース構築・管理支援システム：KDBMS利用の手引
Author(s)	磯本, 征雄
Citation	大阪大学大型計算機センターニュース. 1984, 52, p. 59-86
Version Type	VoR
URL	https://hdl.handle.net/11094/65598
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

データベース構築・管理支援システム

KDBMS 利用の手引

研究開発部 磯 本 征 雄

1. KDBMSの目的とその機能

システムKDBMS^{1,2)}(Knowledge Based Database Management System)は、汎用DBMS^{*}の一つであるINQ³⁾**を使用してデータベース・システムを構築・管理しようとする人に対して、次の4項目について支援と助言をします；

- ①；ジョブデックの自動生成，
- ②；ジョブ実行コマンドの自動発生とその起動，
- ③；ジョブ実行結果の診断，
- ④；上記の処理手順に関する助言・示唆。

本説明書では、このようなKDBMSについて、その利用法を説明します。

KDBMSの利用者は、KDBMSをTSS端末より次のコマンドによって呼び出せます。

SYSTEM? KDBMS

KDBMS呼び出し後のデータベース構築・管理の助言は、対話形式で逐次的に進められます。

INQが十分理解できない人にも、本システムのガイダンスに従えば、データベースの構築・管理が簡単にできます。また、INQを良く知っている人にとっても本システムの活用で、INQ 利用上の操作を簡略化することができます。

さらに、DBMSであるCOOD⁴⁾***で作成されたデータベースを既に持っている利用者のためには、INQへ自動変換できる機能も備えています。⁵⁾

脚注

*……データベース管理システム(Database Management System)の略称で、データベースの構築・運営・管理を行うソフトウェアです。

**……INQとはInformation Queryの略称で、大阪大学大型計算機センターのACOSシステム1000に備わっているDBMSの一種です。

***…Conversational Organized Database Management systemの略です。会話型データベース管理システムの一種で、初心者でも非常に簡単にデータベースが構築でき、小型のデータベースの構築・管理・運営に適しています。

2. データベース構築の手順

KDBMSによるデータベース構築の手順は、図1に示すように、8つのステップから成っています。

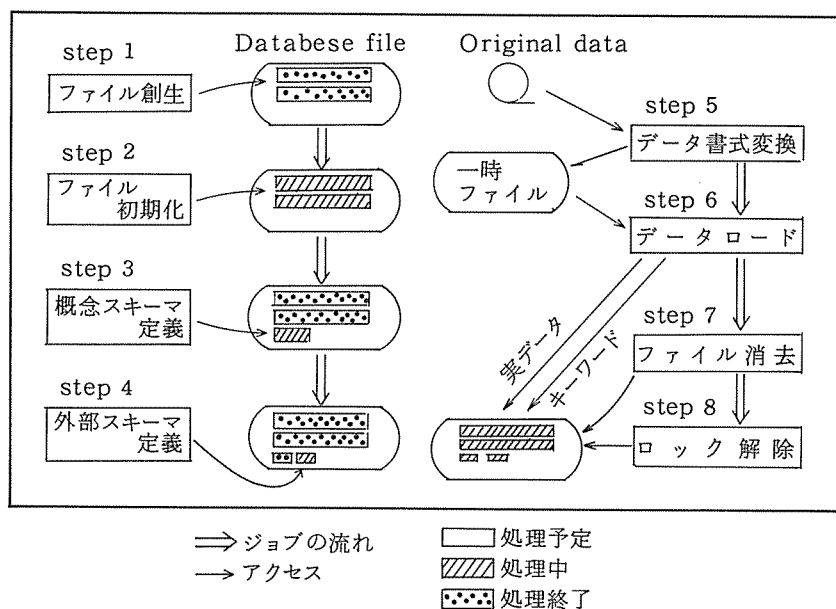


図1 KDBMSによるデータベース構築・管理の手順

〔ステップ1〕 CREATE；

データベース・ファイルの創生のためのジョブ・デックの編集と実行を行います。

〔ステップ2〕 INITIAL；

データベース・ファイルの初期化のためのジョブ・デックの編集と実行を行います。

〔ステップ3〕 CONCEPTUAL；

概念スキーマの定義（データベース・ファイルの論理構造の記述）の登録のためのジョブ・デックの編集と実行を行います。

〔ステップ4〕 EXTERNAL；

外部スキーマの定義（データベース・ファイルよりデータを読み出すときの論理構造の記述）の登録のためのジョブ・デックの編集と実行を行います。

〔ステップ5〕 DATA CONVERSION；

原データを、INQのデータ格納用ユーティリティ・プログラム（INQローダ

と呼ぶ)の要求する書式に変換するプログラムを編集しかつ実行します。

〔ステップ6〕 DATA LOAD;

データベース・ファイルヘデータの格納のためのジョブ・デックの編集と実行を行います。

さらに様々な異常処理に対処するために、KDBMSでは上述の6ステップに付け加え次の処理段階を準備しています。

〔ステップ7〕 DELETE

データベース・ファイルの消去を行います。

〔ステップ8〕 ABORT

これはアボートログ^{*}の解除を行います。

これらの順序関係を図式化したものが図1です。ステップ1～4でデータベースの論理的定義を行い、ステップ5によって原データの書式を変換し、そしてステップ6でデータベース・ファイルヘデータの格納を行っています。

図1に示された各ステップの各々の処理の内訳は、さらに4つの処理モードから成っています。

- 1) ジョブ・デックの編集 (E;EDIT)
- 2) ジョブの実行 (R;RUN)
- 3) ジョブ実行結果の診断 (D;DIAGNOSE)
- 4) ジョブ・デックのリスト出力(L;LIST)

これを表にしたのが表1です。端末より実際に行われる処理内容は、処理モードと処理段階のステ

表1 作業指示記号

〔形式〕 $\begin{array}{c} \underline{XY} \\ \uparrow \quad \swarrow \\ \text{ジョブ・ステップ番号} \\ \text{処理モード} \end{array}$

X ; 処理モード	Y ; ジョブ・ステップ番号
E ; 編集モード	1 ; CREATE
R ; 実行モード	2 ; INITIAL
D ; 診断モード	3 ; CONCEPT
	4 ; EXTERNAL
L ; リストの出力	5 ; DATA CONVERSION
	6 ; DATALOAD
	7 ; DELETE
	8 ; RELESE AN ABORT LOCK
	9 ; HISTORY

(組み合わせはE1～E8, R1～R8, D1～D8, L1～L9までです。)

脚注

*…………データベースに対して許されないアクセスを行った場合、データベースを保護する為にデータベース・ファイルに対してロックをかけ、以後アクセスできなくなります。

ップ番号の組合わせによって指定されることになります。例えばステップ1のCREATE ジョブデックの編集は、“ステップ1のEDIT”なので“E 1”というように表わされます。以後、すべての作業内容は、処理段階のステップ番号と処理モードの組合せで与えることになります。

ジョブ処理の結果、当然データベースの状態が変わります。その様子をジョブステップごとのジョブ処理モードと、データベースの状態変化の関係で図式化したものが、図2です。中心から動径方向につけられた各円弧の番号が、図1のステップ番号（又は、表1のJOB番号）に相当します。そして、円周方向を6分割したものが、データベースの状態を表わしており、次のものからなっています。

(1) 編集途中 (2) 編集完了 (3) 処理途中 (4) 処理完了 (5) 診断途中 (6) ジョブ完了

KDBMSでは、ジョブ処理状況の履歴をこのようなSpiral model図によって記録・管理しています。

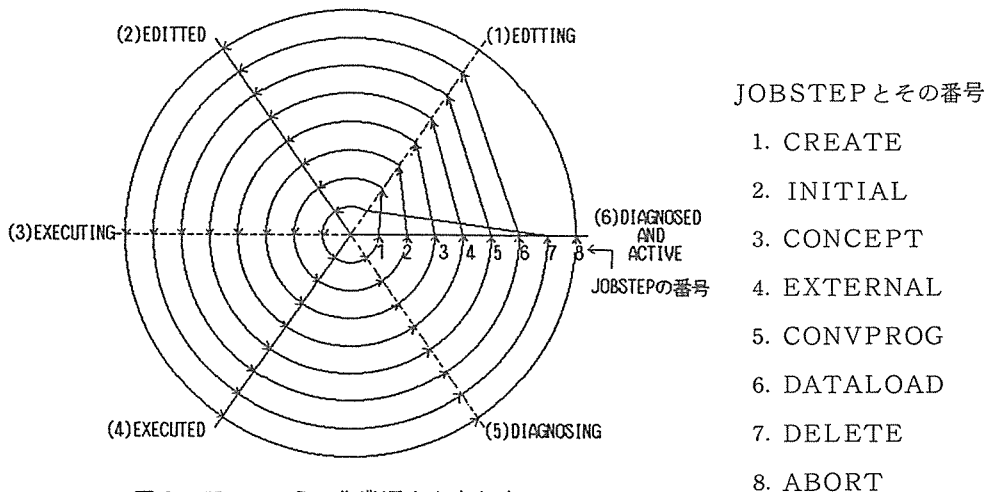


図2 KDBMSの作業順序を表わす
Spiral Model

図2において中心部分の番号1の円周から逐次反時計まわりに矢印に沿って、たどって行くことにより、データベースが構築されてゆく手順を見ることができます。ステップ7の終了後、ステップ1へと実線がSpiralの中心に向かっていているのは、ステップ7でデータベース・ファイルが消去されてしまうために、再度ファイルの創生（CREATE）から実行する必要があるためです。ステップ8はアボートロックの解除であり、機能上他のジョブとは独立していつでもその処理を行えることを示しています。このようなジョブ履歴の内訳はKDBMS呼び出し時に毎回出力され、また利用者の要求によってもいつでも表示されます。さらにKDBMSはこれを使用して、次々とジョブ・デックの処理手順を示唆します。

さて、実際のデータベース構築作業において、システムからKDBMS利用者への助言は、次の

ように与えられます。

```
※ THE NEXT WORK IS 'R1' ※  
IS THIS OK ?
```

```
-----  
OK -----> ( 1 )  
NO -----> ( 2 )  
-----
```

=

この例で示されたクォーテーション・マークにかこまれた記号(R1;CREATE のジョブの実行、すなわちステップ1のRUN)が、次に行うべき作業です。利用者はこれに対して、1(OK)又は2(NO)で答えます。一方、もし手続きが順調に進まずに何らかの誤った操作をした場合や、複雑な判断が要求される場合にはKDBMSがしかるべき助言を与えますので安心して利用できます。

3. KDBMSの支援によるデータベース構築・管理の手順

ここではKDBMSの支援を受けながらデータベースを構築・管理して行く手順を具体例に沿って説明します。以下、図の矢印に沿って手順を見て下さい。図中の下線の付された文字列は、利用者の入力です。

〔I〕 データベース構築の支援

SYSTEM ?KDBMS ← KDBMSの呼び出し

```
*****
* WELCOME TO KDBMS *
*****
```

```
USERID$PASSWORD FOR THE DATABASE ADMINISTRATOR ?
=6000810003$PASSWORD 利用者の課題番号とパスワード
UMC OF THE DATABASE ?
=6000810003 ←データベースが格納されるユーザマスタカタログ名
DATABASE NAME ?
=TESTDB ←データベース名 (通常USERIDと同一視される)
FDL AND ITS NUMBER ?
=TESTFILE 01 ←ファイル名(FDL名)と番号
```

IS 6000810003/TESTDB/TESTFILE OLD OR NEW ?

```
-----
OLD --->( 1 )
NEW --->( 2 )
-----
```

*** ANSWER WITH A NUMERAL (1 OR 2) ***

=1 又は 2を入力

(1)の場合

既にKDBMSを用いてINQファイル
を構築した、又は構築途中である
場合

(2)の場合

データベースを新規構築開始す
る場合

ファイル作成状況のメッセージ (65頁参照)

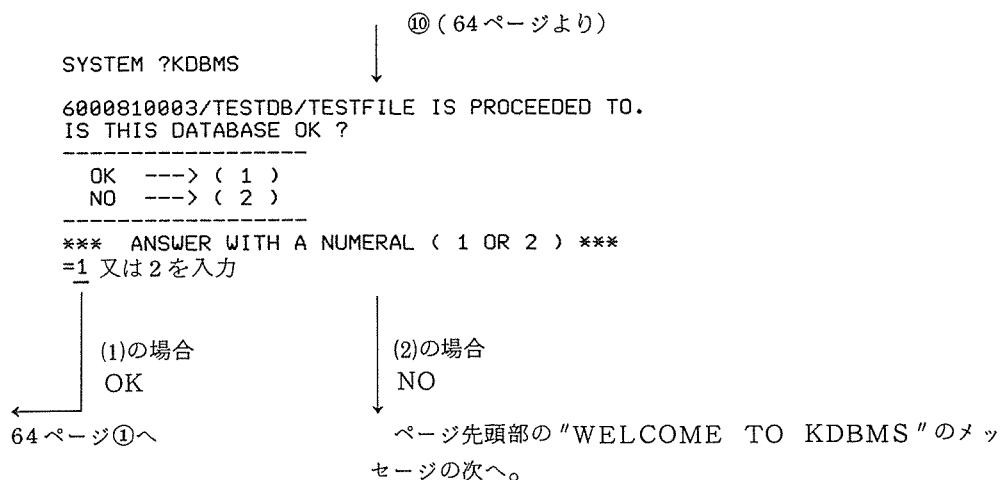
TYPE IN INQ-SECTION NAME OF TESTFILE
=TESTSEQ ← INQセクション名を入力する

=== THE LAST JOB HISTORY ===

CREATE	EDITTING	12/22/83	12.036
INITIAL	EDITTING	12/22/83	12.036
CONCEPT	EDITTING	12/22/83	12.036
EXTERNAL	EDITTING	12/22/83	12.036
CONVPROG	EDITTING	12/22/83	12.036
DATALOAD	EDITTING	12/22/83	12.036
DELETE	EDITTED	12/22/83	12.036
ABORT	EDITTED	12/22/83	12.036

→ 66ページ②へ

(前図においてKDBMSが異常終了した場合の再開)



(ファイル作成状況のメッセージ)

データベース作成のためには、12の補助ファイルが必要です。以下のものは、データベース構築開始時にKDBMSが自動作成したファイル出力例です。各ファイルには、作業履歴やジョブ・デックが格納されます。

INITIALIZATION OF DATABASE CONSTRUCTION

LIST OF CREATED FILES	
ACCE FC 6000810003/TESTDB/INQLIB	← External Schema 格納用ファイル
ACCE FC 6000810003/TESTDB/CATA01/HISTORY	← 履歴記録用ファイル
ACCE FC 6000810003/TESTDB/CATA01/CREATE	← データベース・ファイル生成用
ACCE FC 6000810003/TESTDB/CATA01/INITIAL	← ジョブ・デックのファイル
ACCE FC 6000810003/TESTDB/CATA01/CONCEPT	← データベース・ファイル 初期化用
ACCE FC 6000810003/TESTDB/CATA01/EXTERNAL	← ジョブ・デックのファイル
ACCE FC 6000810003/TESTDB/CATA01/CONVPROG	← Conceptuel Schema 定義用
ACCE FC 6000810003/TESTDB/CATA01/DATALOAD	← ジョブ・デックのファイル
ACCE FC 6000810003/TESTDB/CATA01/DELETE	← External Schma 定義用ジョブ・デックのファイル
ACCE FC 6000810003/TESTDB/CATA01/ABORT	← 原データ書式変換用ジョブ・デックのファイル
ACCE FC 6000810003/TESTDB/CATA01/JOUTFILE	← データ格納用ジョブ・デックのファイル
ACCE FC 6000810003/TESTDB/CATA01/OPTI	← データベース・ファイル消去用ジョブ・デックのファイル
*** END OF FILE CREATION ***	

(64 ページより) ②

(76 ページより)

HOW WILL YOU HAVE ACCESS TO THIS DATABASE ?

CONSTRUCTION ----> (1)

MAINTENANCE ----> (2)

RETRIEVAL -----> (3)

*** ANSWER WITH A NUMERAL (1, 2, OR 3) ***

=1 又は 2 又は 3 を入力

(1)の場合

構築

(2)の場合

管理・更新

(3)の場合

検索

→ 74 ページへ

→ 76 ページへ

HOW WILL YOU DECIDE THE JOB SEQUENCE,

ACCORDING TO KDBMS'S SUGGESTION --> (1)

BY YOURSELF -----> (2)

** ANSWER WITH A NUMERAL (1 OR 2). **

=1 又は 2 を入力

=====

LET'S BEGIN.

(1)の場合

KDBMSからの
作業指示を受ける

(2)の場合

利用者が自分で次の作業を決定する場合

→ 73 ページ②へ

* THE NEXT WORK IS 'E1' *
IS THIS OK ?

OK ----> (1)

NO ----> (2)

=1 又は 2 を入力

(1)の場合

E 1 を行う

(2)の場合

利用者による作業選択

→ 73 ページ②へ

!!! EDITTING CREATE JUST NOW !!!

* HOW WILL YOU EDIT THE JOB DECK OF 'CREATE ' ? *

AUTOMATIC MODE -----> (1)

EDIT BY YOURSELF ----> (2)

*** ANSWER WITH A NUMERAL (1 OR 2) ***

=1 又は 2 を入力

(1)の場合

自動編集モードを使用する

→ 67 ページ③へ

(2)の場合

利用者による編集・修正

→ 73 ページ⑥へ

(66 ページより) ③

GENERATION OF 'CREATE'

SIZE OF THE DATABASE ? (LINKS)

=10 ← データベースの大きさ(リンク単位で)を入力

*** 'CREATE' JOB DECK HAS BEEN GENERATED ***

!!! EDITTED CREATE JUST NOW !!!

* THE NEXT WORK IS 'R1' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

= 1 又は 2 を入力

(1)の場合

R 1 を行う

(2)の場合

R 1 を行わない

→ 73 ページ ② へ

!!! RUNNING CREATE JUST NOW !!!

*** START THE BATCH JOB FOR CREATE ***

SNUMB # H730T

!!! RUN CREATE JUST NOW !!!

EXECUTING THE JOB FOR CREATE

JMON *

H730T -001 EXECUTING @ 10:13:09

H730T IN DEMAND-FILE OUTPUT WAITING ID=G7 @ 10:13:22

NORMAL TERMINATION

=

END OF EXECUTION

COUT

*** ARRANGEMENT OF JOUT FILE ***

JOUT H730T

FUNCTION?LIST ALL

ACTI#	RC	LINE	HOLD	CLASS
--	\$\$	--	--	C
001	12	6	YES	C

FUNCTION?HOLD

=

CONTINUE DIAGNOSIS.

COUT

WAIT A MINUTE WHILE THE COPY OF JOUTFILE

JOUTFILE HAS BEEN COPIED.

COUT

*** END OF THE ARRANGEMENT ***

!!! DIAGNOSD CREATE JUST NOW !!!

=== START THE DIAGNOSIS ===

*** THERE IS NO ERROR. *** ← 診断結果

!!! ACTIVE CREATE JUST NOW !!!

→ 68 ページ ④ へ

実行中のシステム
からのメッセージ

ジョブ実行結果の
診断のためのシス
テムからのメッセ
ージ

(67 ページより) ④

* THE NEXT WORK IS 'E2' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

=1 又は 2 を入力

(1)の場合
E 2 を行う

(2)の場合
E 2 を行わない

→ 73 ページ④参照

GENERATION OF 'INITIAL'

*** 'INITIAL' JOB DECK HAS BEEN GENERATED ***

!!! EDITED INITIAL JUST NOW !!!

* THE NEXT WORK IS 'R2' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

=1 又は 2 を入力

(1)の場合
R 2 を行う

(2)の場合
R 2 を行わない

→ 73 ページ④参照

実行及び診断
(CREATE の場合と同様)

* THE NEXT WORK IS 'E3' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

=1 又は 2 を入力

(1)の場合
E 3 を行う

(2)の場合
E 3 を行わない

→ 69 ページ⑤へ

→ 73 ページ④参照

(68 ページより) ⑤

↓
GENERATION OF 'CONCEPT'

---- HOW DO YOU EDIT THE FDL ? ----

COOD -----> (1)

FORTRAM PROGRAM -----> (2)

EDIT UNDER ASSISTANCE OF FDL EDITOR --> (3)

EDIT ALL BY YOUR SELF -----> (4)

原データの格納形式について

*** ANSWER WITH A NUMERAL (1, 2, 3, OR 4) ***

=1 ←この場合COODであることを宣言している。(その他の場合、2は付録1を、4は付録2をそれぞれ参照して下さい。)

CATA/FILE FOR DDL OF COOD DATABASE ?

=6000810003/BADGE-X/DDDL-5701 ← COODデータベースのDDLが収められている

ファイル名

TABLE NAME ? ←COODのテーブル名

=KOZIN

COOD CONCEPT SCHEMA IS ANALYSED.

--- START THE GENERATION OF FDL ---

PASSWORD ?

=2 ←INQファイルに対してパスワードを与える
(キャリッジリターンの場合はパスワード無しとする。)

*** CONCEPTUAL SCHEMA HAS BEEN GENERATED ***

!!! EDITED CONCEPT JUST NOW !!!

↓
* THE NEXT WORK IS 'R3' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

=1 又は 2 を入力

(1)の場合
R 3 を行う

(2)の場合
R 3 を行わない

→73ページ④参照

実行及び診断 (CREATE の時と同様)

↓
* THE NEXT WORK IS 'E4' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

=1 又は 2 を入力

(1)の場合
E 4 を行う

(2)の場合
E 4 を行わない

→70ページ⑥へ

→73ページ④参照

(69 ページより) ⑥



GENERATION OF 'EXTERNAL'

利用者は何も入力しなくてよい。

*** 'EXTERNAL' JOB DECK HAS BEEN GENERATED ***

!!! EDITED EXTERNAL JUST NOW !!!



* THE NEXT WORK IS 'R4' *
IS THIS OK ?

-----> (1)
OK -----> (2)
NO -----> (2)

=1 又は 2 を入力

(1)の場合
R 4 を行う

(2)の場合
R 4 を行わない
→ 73 ページ@参照

実行及び診断
(CREATEと同様)



* THE NEXT WORK IS 'E5' *
IS THIS OK ?

-----> (1)
OK -----> (2)
NO -----> (2)

=1 又は 2 を入力

(1)の場合
E 5 を行う

(2)の場合
E 5 を行わない
→ 73 ページ@参照

→ 71 ページ⑦へ

(70 ページより) ⑦

GENERATION OF DATA CONVERSION PROGRAM.

--- HOW WILL YOU GENERATE THE DATA CONVERSION PROGRAM ? ---
WITH COOD FORMAT -----> (1)
WITH FORTRAN PROGRAM -----> (2)
EDIT BY YOURSELF -----> (3)
AN FDL HAS ALREADY GIVEN ----> (4)
*** ANSWER WITH A NUMERAL (1, 2, 3, OR 4) ***
=1 ← ここでは原データの格納形式がCOODであることを指定している。
TABLE NAME ? ← その他の場合は付録を参照して下さい。
=KOZIN ← COODのテーブル名
TOTAL RECORD COUNTS ? ← 書式変換するレコードの件数
=150
*** DATA CONVERSION PROGRAM HAS BEEN GENERATED ***
!!! EDITED CONVPROG JUST NOW !!!

* THE NEXT WORK IS 'R5' *
IS THIS OK ?

OK -----> (1)
NO -----> (2)

=1 又は 2

(1)の場合
R 5 を行う

(2)の場合
R 5 を行わない
→ 73 ページ@参照

実行及び処理
(CREATE の時と同様)

→ 72 ページ⑧へ

(71 ページより) ⑧

* THE NEXT WORK IS 'E6' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

=1 又は 2

(1)の場合
E 6を行う

(2)の場合
E 6を行わない
→ 73 ページ④参照

GENERATION OF 'DATALOAD' JOB DECK

TOTAL RECORD COUNTS?

=150

*** 'DATALOAD' JOB DECK HAS BEEN EDITED ***

!!! EDITED DATALOAD JUST NOW !!!

* THE NEXT WORK IS 'R6' *
IS THIS OK ?

OK -----> (1)

NO -----> (2)

=1 又は 2

(1)の場合
R 6を行う

(2)の
R 6を行わない
→ 73 ページ④参照

実行及び処理
(CREATEのときと同様)

完 成

以上で、データベース構築は完了します。この後、KDBMSは66頁②へと誘導します。以後では補足的事項について解説します。

④〔利用者が次に行う作業を自分で決定する場合〕

```

SELECT ONE OF THE NUMERICS OF JOBS AND MODES .
=====
(MODE) E;EDIT  R;RUN  D;DIAG  L;LIST  J;JMONI  F;FINISH
=====
(JOB) (1)CREATE      (2)INITIAL    (3)CONCEPT
      (4)EXTERNAL    (5)CONVPROG   (6)DATALOAD
      (7)DELETE      (8)ABORT      (9)HISTORY
=====
=

```

この表が出力された時点で利用者はモードとジョブステップの組合せによる作業の入力を行って下さい。この場合一度に複数の作業を選択でき、入力順に処理されます。入力の際はコマンドとコマンドとの間を“ , ”で区切って下さい。

例

E 1, E 2, R 1, R 2

⑤〔利用者自身によるジョブデックの編集及び修正 (エディター・サブシステムの利用)〕

ジョブデックの編集はモードE (EDIT)によります。そこでEDIT BY YOURSELF を選んだ場合はKDBMSによって、自動的にACOS-6. EDITORサブシステムが呼ばれ、編集 (修正)の対象となっているジョブデックの格納されているファイルが自動的に呼び出されます。利用者は必要に応じて各種コマンド (EDITORサブシステム)を用い編集、又は修正を行って下さい。そして最後に必ず、RESAVE△* (△はスペースを示す)を行い、DONEを入力して下さい。その他の事柄については、KDBMSより助言しますので、それに従って下さい。

例：以下EDITORサブシステム稼動例

```

:
-RESA△* (△はスペースを示す)
DATA-SAVED
-DONE
}
KDBMSへ復帰する

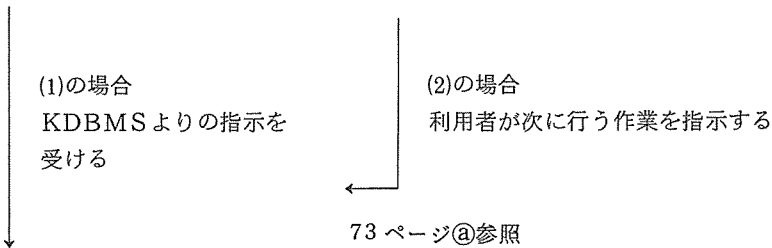
```


〔Ⅱ〕 データベースの更新と管理の支援

既に完成しているデータベースの更新を支援する場合の例を示します。

```
=== THE LAST JOB HISTORY ===
CREATE      EDITTING 12/22/83 12.036
INITIAL     EDITTING 12/22/83 12.036
CONCEPT   EDITTING 12/22/83 12.036
EXTERNAL    EDITTING 12/22/83 12.036
CONVPROG    EDITTING 12/22/83 12.036
DATALOAD    EDITTING 12/22/83 12.036
DELETE      EDITTED  12/22/83 12.036
ABORT       EDITTED  12/22/83 12.036

HOW WILL YOU HAVE ACCESS TO THIS DATABASE ?
CONSTRUCTION ---> ( 1 )
MAINTENANCE ----> ( 2 )
RETRIEVAL -----> ( 3 )
*** ANSWER WITH A NUMERAL ( 1, 2, OR 3 ) ***
=2 ←更新・管理であることを示す,
HOW WILL YOU DECIDE THE JOB SEQUENCE,
ACCORDING TO KDBMS'S SUGGESTION --> ( 1 )
BY YOURSELF -----> ( 2 )
** ANSWER WITH A NUMERAL ( 1 OR 2 ) **
=1 又は 2 を入力
```



```
YOUR DATABASE HAS ALREADY BEEN CONSTRUCTED

WHICH WILL YOU UPDATE IN YOUR DATA ? ← どの部分を変更するのか？
(1) CONTENTS OF THE STORED DATABASE. ← 格納データの更新
(2) THE CONCEPTUAL SCHEMA. ← 概念スキーマの再定義
(3) THE FILE SIZE. ← データベース・ファイルの大きさの変更
.....
SELECT ONE OF THE NUMERALS.
=3 又は 1 又は 2 を入力

(1), (2), (3)
→ 75 ページ⑨へ
```

(74 ページより) ⑨

↓

THERE IS(ARE) 2-CANDIDATE NEXT WORK(S).
ANSWER THE QUESTION(S) (0.0-1.0).
HAVE YOU UPDATE THE PAGE SIZE(OR FILE SIZE)?
=1.0 ← 利用者の判断 (1.0 から 0.0 まで) 脚注 1
WILL YOU UPDATE THE CONCEPTUAL SCHEMA?
=0.0

*** CONCLUSION ***

.....
(1) EDIT OF CREATE ;E1
LET'S EDIT THE JOB DECK 'CREATE',
BECAUSE THE PREVIOUS WORK WAS MISTAKEN.

} KDBMS よりの指示

SELECT ONE OF THE NUMERALS (1-1)

= 1 または (C/R) を入力

←

(1) の場合 (脚注 2 参照)
KDBMS よりの指示に従い
この場合 E 1 を行う

(C/R) (キャリッジリターン) の場合
KDBMS よりの指示には従わず、利
用者が判断する。

66 ページ中程の E 1 へ

73 ページ④参照
→ 作業終了後、再び⑨へ

脚注 1

(1.0~0.0 の返答方式について)

KDBMSでは、YESを1.0、NOを0.0とし0.1きざみで11段階で、あいまいさをもった表現の返答ができます。1.0から0.0にYESからNOへとその程度を数値で表し、これを質問の次に入力して下さい。

脚注 2……KDBMSよりの示唆が複数の場合はその中のどれか1つを選択して入力する。又は、キャリッジリターンを入力する。

〔Ⅲ〕 データベース検索の支援

完成したデータベースで検索する場合の例を示します。

```
=== THE LAST JOB HISTORY ===
CREATE      ACTIVE      12/20/83 12.056
INITIAL     ACTIVE      12/20/83 12.462
CONCEPT   ACTIVE      12/20/83 13.202
EXTERNAL    ACTIVE      12/20/83 13.255
CONVPROG    ACTIVE      12/21/83 10.556
DATALOAD    ACTIVE      12/21/83 11.455
DELETE      EDITTED     12/19/83 20.308
ABORT       EDITTED     12/19/83 20.308

HOW WILL YOU HAVE ACCESS TO THIS DATABASE ?
CONSTRUCTION ---> ( 1 )
MAINTENANCE ----> ( 2 )
RETRIEVAL -----> ( 3 )
*** ANSWER WITH A NUMERAL ( 1, 2, OR 3 ) ***
= 3 データベース検索であることを示す。
```

=====

LET'S BEGIN.

INQ EQL/JIPS VERSION 9.1-00 22:56:19 12-22-1983

```
-----
: FILE NO : FILE NAME      : RECORD CNT : DATABASE NAME :
:   01   : TESTFILE       :    245    : TESTDB        :
-----
```

```
-----
: INQSECTION NAME : TYPE : INQ FILE NO :
: TESTSEQ         : 01   :    01       :
-----
```

*** RETRIEVAL START OF YOUR DATABASE ***
? ____ ← EQL コマンド

.
.
.
} 利用者による検索 (EQL マニュアル参照)

?DONE ← EQL 終了コマンド
*** END OF KDBMS'S RETRIEVE ***

└─┘
←
66 ページ②へ

〔EQLについて〕

このDBMSでは、検索方法として INQ /EQL（会話型エンドユーザ言語）を使用しています。
第2表に主なコマンドを示します。

表2 主なEQLコマンド

EQL コマンド	意 味 と 使 用 例
RETRIEVE	検索 (例)? RETRIEVE┐〈データ項目名〉┐EQ┐パラメータ
FIELD	FDLの表示 (例)? FIELD
SAVE	概に検索したレコードセットを対象としてデータをファイルにセーブする。 (例)? SAVE┐1
AND, OR, NOT	SAVEファイル同志の論理積、論理和、否定を求める。 (例)? AND┐1┐2
SORT	SAVEファイルの1つのデータ項目を対象として、降順または、昇順にソーティングする。 (例)? SORT┐1
DISPLAY	検索又は、SAVEファイルの内容を指定したデータ項目を対象として表示する。 (例)? DISPLAY 〈データ項目名〉┐・・・┐〈データ項目名〉┐・・・
DONE	検索を終了する。 (例)? DONE

4. ジョブ実行後の診断でエラーが検出された場合について

エラーが検出された場合、KDBMSより、例えば次のような診断結果が出力されてきます。

```

=== START THE DIAGNOSIS ===
*** THERE IS (ARE) 1 ERROR MESSAGE(S). ***
*** ERROR-CODE = ***
      TOO LARGE SIZE OF THE DATABASE FILE.
```

この診断結果の例は、CREATEのジョブの実行を行った時に、指定したファイル・サイズが大きすぎ、UMCに登録されたファイル容量を超えたために出力されたエラーです。このエラー出力に続けて、KDBMSはさらに次のように質問してきます。

```

THERE IS(ARE) 2-CANDIDATE NEXT WORK(S).
=== LET'S DETERMINE THE NEXT WORK ===
ANSWER THE QUESTION(S) (0.0-1.0).
CAN YOU DECREASE THE MASS OF LOADED DATA?
=1.0
WILL YOU REQUIRE THE COMPUTER CENTER MORE FILE SIZE?
=0.0

```

エラーが検出された場合、そのJOBの実行は中断されることになり、そのステップの処理状況を表わす履歴はEXECUTINGからEDITTEDと変わります。そして、このエラー処理方法についての問診が行われた後、次に行うべき作業が決定され、エラーの処理に入ります。いずれの場合においても、利用者は1.0 (YES)～0.0 (NO)などの数字で同意の程度を回答して下さい。これに応じてKDBMSは次の助言を適切に与えます。

```

*** CONCLUSION ***
.....
(1) EDIT OF CREATE ;E1          ← 次の作業
LET'S EDIT THE JOB DECK 'CREATE', } ← 次の作業の説明およびその理由
BECAUSE THE PREVIOUS WORK WAS MISTAKEN.
*****

```

5. む す び

KDBMSは、INQの活用の際して、利用者を支援して行くエキスパート・システムです。本システムを使用することによって、

1. データベースのためのファイル領域確保
2. 概念スキーマの入力の支援
3. データ書式変換プログラムの一部コーディング
4. JCLの自動生成
5. ジョブ実行の代行
6. 処理の全体の流れの示唆

が支援されるので、データベース・システム構築・管理が非常に容易になります。従来、非専門家としてのデータベース管理者にとっては、汎用データベース管理システムを活用することは大きな負担でした。本システムの活用がこのことに関する問題を解決してくれる事を願っています。

KDBMSは、データベース構築支援のTOOL提供をめざして研究開発されたものです。すなわち、本システムは、本センター、研究開発部の活動として開発され、試験的なサービスを行うものです。もし問題点や不十分な点がありましたら御指摘して下さい幸いです。本システムの利用についての御質問は、大阪大学大型計算機センター磯本征雄（内線 2833）まで御連絡をお願いします。最後に本システムの整備にあたり、様々の御協力をいただきました大阪電気通信大学内木良彰君に感謝いたします。

参 考 文 献

- 1) 磯本征雄, 石桁正士, 溝口理一郎, 角所収 ; データベース構築・管理のための知的支援システム—Knowledge Based DBMS (KDBMS)—, 情報処理学会「アドバンスト・データベース・システム」(1982, 12月), pp.49-58.
- 2) Yukuo ISOMOTO, Tadashi ISHIKETA, Riichiro MIZOGUCHI, and Osamu KAKUSHO ; Knowledge Based DBMS for Non-professional Database administrators, Proceedings of COMP CON FALL '83, "Delivering Computer Power to End Users", IEEE, (1983), pp.514-522.
- 3) 日本電気 ; ACOSオペレーティングシステム, ACOS-6データ管理, INQ文法説明書, INQ運用説明書.
- 4) 松田孝子, 田中信行 ; 会話型データベース管理システム, COOD説明書, 文部省科研費特定研究(1)「トレース・キャラクターゼーションに関する情報の処理, 収録, 管理, 流通に関する研究」
- 5) Yukuo ISOMOTO, Takako MATSUDA, and Nobuyuki TANAKA ; Guidance System for Structuring or Restructuring of a Database in Multiple Database Management Systems, Journal of Information Processing, Vol.5, No.3 (1982), pp.182~187.

付録.1 概念スキーマの編集について

INQでは、概念スキーマのことをFDL (File Description Language) と呼びます。この付録.1では“E3”すなわちCONCEPTジョブデックの編集時において、与えられるオプション；

```
----- HOW DO YOU EDIT THE FDL ? -----
COOD -----> (1)
FORTRAN PROGRAM -----> (2)
EDIT UNDER ASSISTANCE OF FDL EDITOR -> (3)
EDIT ALL BY YOURSELF -----> (4)
```

で原データの書式が“COOD”以外の形式で与えられている場合についてそれぞれの処理内容を説明します。

(1) 2番のFORTRAN PROGRAMを指定した場合

ここで言う“FORTRAN PROGRAM”とは、次に示しているようなDOループを用いてREAD文で原データを読み、WRITE文で端末へ出力するプログラムのことをいいます。ただし、入力ファイルはATTACH*命令によりファイルコード08番以外の番号を指定して下さい。

```
CHARACTER DATA1*10,DATA2*10,DATA3*10
CALL ATTACH(01,'TESTDATA;',3,1,ISTAT,)
DO 111 N=1,100
  READ(01,100,END=999)DATA1,DATA2
  WRITE(06,100)DATA1,DATA2
100 FORMAT(2A10)
DO 222 L=1,10
  READ(01,200)DATA3
  WRITE(06,200)DATA3
200 FORMAT(A10)
222 CONTINUE
111 CONTINUE
999 STOP
END
```

データ入力の繰り返しは、必ずDOループを使ってコーディングして下さい。詳細は、付録2に説明されるものに同じです。

次に、上記FORTRANプログラムをもとにして、編集モードでFDLのためのジョブデックの編集手順と、編集されたジョブデックを示します。

脚注

*……………パーマネント・ファイルをアクセスするのに用いる、FORTRAN基本外部サブルーチンです。その一般形式は次のようになります。

```
CALL ATTACH(u, “カタログ/ファイル名;”, P, n, ISTAT,)
u ; ファイルコード P ; パーミッション ( P=1 ; READ P=2 ; WRITE
P=3 ; READ, WRITE ) n ; モード ( n=1 ; 順編成 n=2 ; 直接編成 )
```

```

----- HOW DO YOU EDIT THE FDL ? -----
      COOD -----> (1)
      FORTRAN PROGRAM -----> (2)
      EDIT UNDER ASSISTANCE OF FDL EDITOR -> (3)
      EDIT ALL BY YOURSELF -----> (4)
*** ANSWER WITH A NUMERAL (1, 2, 3, OR 4) ***
=2 ← “FORTRAN PROGRAM”を選択している。

CATA/FILE OF THE PROGRAM FOR DATA CONVERSION?
=6000810003/TESTPROG ←FORTRAN PROGRAMが格納されている
                                     カタログ/ファイル名
GIVE THE GROUP NAME TO THE FOLLOWING DATA ITEMS.
-----
      WRITE(06,200)DATA3
-----
=GDAT3 ←点線内に示されているデータ項目名につけるグループ名

THIS PROGRAM IS ANALIZED.

--- START THE GENERATION OF FDL ---

PASSWORD ?
=7 ←パスワードの指定
*** CONCEPTUAL SCHEMA HAS BEEN GENERATED ***

```

以上の手順によって、以下のジョブ・デックが生成されます。（元のFORTRANプログラムは付録2に示す。）

```

##S,J N
$      JOB      6000810003$PASSWORD
$      PROGRAM  FDL
$      PRMFL    **,R,R,INQ/FDL
$      PRMFL    H*,R,R,INQ/FDL
$      FILE     S1,X1D,50R
$      FILE     S2,X2D,50R
$      FILE     S3,X3D,50R
$      FILE     I1,X4D,50R
$      FILE     I2,X5D,50R
$      SYSOUT   LP,ORG
$      PRMFL    DB,W,R,6000810003/TESTDB/TESTFILE
$      DATA    CD
CREATE
FDL TESTFILE,01.
DATABASE TESTDB.
      02  DATA1          PIC X(10).
      02  DATA2          PIC X(10).
      02  GDAT3 (N).      .
      03  DATA3          PIC X(10).
END

```

(2) 3番の“EDIT UNDER ASSISTANCE OF FDL EDITOR”を指定した場合
これを指定した場合、次に示しているようにパスワードの入力に引き続き、FDLについてのコメントが出力され、利用者はFDLを一行ずつ自分で入力することになります。


```

----- HOW DO YOU EDIT THE FDL ? -----
COOD -----> (1)
FORTRAN PROGRAM -----> (2)
EDIT UNDER ASSISTANCE OF FDL EDITOR -> (3)
EDIT ALL BY YOURSELF -----> (4)
*** ANSWER WITH A NUMERAL (1, 2, 3, OR 4) ***
= 3 ← 3による方法を指定している

* START THE EDITION OF AN FDL *

PASSWORD ?
= 2 ← パスワードの指定
ENTER FDL LIKE THIS.

02 DATANAME1 PIC 9(2) PKY. ----- PRIMARY KEY
02 DATANAME2 PIC X(12) DSP.
02 DATANAME3 (N). ----- REPEATING GROUP
   03 DATANAME4 PIC FB. ----- FLOATING BINARY
   03 DATANAME5 PIC CP6. ----- 1 WORD BINARY
99 DN1 = DATANAME1. ----- OTHER DATA NAME
99 DN2 = DATANAME2. ----- OTHER DATA NAME

=02 SEQNO PIC CP6. }
=02 CODENO PIC X(3). } 利用者によるFDLの入力
=02 HOWKY (N). }
= 03 HOW PIC X(12). }
= 2

*** CONCEPTUAL SCHEMA HAS BEEN GENERATED ***

```

KDBMSは入力されたFDL記述について文法チェックを行います。FDL入力の際には、語と語の区切りには必ず、スペースを開けて下さい。また、FDLを全て入力した後に最後の行で、キャリッジ・リターンのみを入力することにより、CONCEPTジョブデックの編集は終了します。より詳しいFDL記述の文法については、INQ文法説明書を参照して下さい。

付録.2 データ書式変換プログラムの編集について

ここでは“E5”すなわちデータ書式変換プログラムの編集において、与えられるオプション；

```
--- HOW DO YOU GENERATE THE DATA CONVERSION PROGRAM ? ---  
WITH COOD FORMAT -----> (1)  
WITH FORTRAN PROGRAM -----> (2)  
EDIT BY YOURSELF -----> (3)  
AN FDL HAS ALREADY GIVEN --> (4)  
*** ANSWER WITH A NUMERAL (1, 2, 3, OR 4) ***  
=____
```

で原データの書式が“COOD”以外の形式で与えられている場合についてそれぞれの処理内容を説明します。

(1) 2番の“WITH FORTRAN PROGRAM”を指定した場合

このFORTRAN PROGRAMは付録.1のFORTRAN PROGRAMと同じもので、DOループを用いてREAD文で原データを読み、WRITE文で端末へ出力するプログラムのことをいいます。この原データの入出力のためのFORTRANプログラム作成にあたり、次のことに注意して下さい。

- 1 ; 原データが格納されているファイルをアクセスする場合、必ずATTACH文を用いること。
- 2 ; 行番号はSTRIPしておくこと。
- 3 ; ファイルコード08は使用しないこと。
- 4 ; CONCEPTジョブデック編集時に指定したFORTRAN PROGRAMと同一のものであること。

それでは、編集の実例としてFORTRAN PROGRAMの一例と、その処理例を示します。

〔原データ入出力用FORTRANプログラム例〕

以下のプログラムは、原データを読み、これを端末に出力します。KDBMSはこれを解読しFDL及びデータ書式変換プログラムを自動生成します。

```
CHARACTER DATA1*10,DATA2*10,DATA3*10  
CALL ATTACH(01,'TESTDATA;',3,1,ISTAT,)  
DO 111 N=1,100  
  READ(01,100,END=999)DATA1,DATA2  
  WRITE(06,100)DATA1,DATA2  
100 FORMAT(2A10)  
  DO 222 L=1,10  
    READ(01,200)DATA3  
    WRITE(06,200)DATA3  
200 FORMAT(A10)  
222 CONTINUE  
111 CONTINUE  
999 STOP  
END
```

〔 KDBMSによって上記FORTRANプログラムを再編集している例 〕

前記FORTRANプログラムは、次の手順を経てデータ書式変換プログラムに再編集されます。

```
--- HOW DO YOU GENERATE THE DATA CONVERSION PROGRAM ? ---  
WITH COOD FORMAT -----> (1)  
WITH FORTRAN PROGRAM -----> (2)  
EDIT BY YOURSELF -----> (3)  
AN FDL HAS ALREADY GIVEN --> (4)  
*** ANSWER WITH A NUMERAL (1, 2, 3, OR 4) ***  
=2 ← FORTRANプログラムの再編集を指定している。  
  
THIS PROGRAM IS ANALIZED.  
  
*** A DATA CONVERSION PROGRAM HAS BEEN GENERATED ***
```

!!! EDITTED CONVPROG JUST NOW !!!

(2) 3番の“AN FDL HAS ALREADY BEEN GIVEN”を指定した場合

データベース構築用ジョブデックを編集する際に、原データについては未定であり、FDLのみが分かっている場合もしばしばあります。この場合、KDBMSはデータ書式変換プログラムのFDL関連部のみをコーディングし、原データの入力の部分については未定のままに残しておきます。つまり、原データからのデータ入力（READ文）については、利用者自身でコーディングすることが必要です。

次に示すFDLを持つデータベースに対する、データ書式変換プログラムの編集を具体例として説明します。

```
02 NAME      PIC X(50) PKY.  
02 CODENO    PIC CP6.  
02 SKILLS (N).  
    03 SKILL   PIC X(20).
```

以下に示している例は、上記FDLに対応してKDBMSによって生成されたデータ書式変換プログラムの出力例と、それに引きついでEDITORで編集を行っている実例です。後半の部分は、データ書式変換プログラムに“READ文”が追加された後の完成プログラムのリストが示されています。

GENERATION OF DATA CONVERSION PROGRAM.

--- HOW DO YOU GENERATE THE DATA CONVERSION PROGRAM ? ---

WITH COOD FORMAT -----> (1)

WITH FORTRAN PROGRAM -----> (2)

EDIT BY YOURSELF -----> (3)

AN FDL HAS ALREADY GIVEN --> (4)

*** ANSWER WITH A NUMERAL (1, 2, 3, OR 4) ***

=4

THE NUMBER OF RECORD SETS ?

=50

EDIT A DATA CONVERSION PROGRAM

BECAUSE READ-STATEMENTS FOR RAW DATA ARE LEFT TO BE COMMENT

-LIST

00010##S,J N

00020\$ JOB 6000810003\$PASSWORD,B

00030\$ LIMITS 25,,,6000

00040\$ LOWLOAD

00050\$ OPTION FORTRAN,RELMEM

00060\$ FORTRAN LISTIN,BIN,NLNO,NFORM

00070CCC FORTRAN PROGRAM FOR DATA CONVERSION CCC

00080COMMENT : INSERT YOUR PROCEDURE.

00090C*** DECLARATION FOR VARIABLES ***

00100 CHARACTER NAME*50,SKILL*20

00110 INTEGER CODENO

00120C*** END OF DECLARATION ***

00130COMMENT : INSERT YOUR PROCEDURE.

00140C*** UNIQUE ITEMS ***

00150 DO 7001 I01=1, 50

00160COMMENT : INSERT YOUR PROCEDURE.

00170 IF(NAME.EQ.' ') GO TO 8000

00180 IREC=01

00190 WRITH(08) IREC,NAME,CODENO

00200C*** GROUP DNAME IS SKILLS ***

00210 DO 7002 I02=1, 0

00220COMMENT : INSERT YOUR PROSEDURE.

00230 IF(SKILL.EQ.' ') GO TO 7102

00240 IREC=02

00250 WRITE(08) IREC,SKILL

00260 7002 CONTINUE

00270 7102 CONTINUE

00280 7001 CONTINUE

00290COMMENT : INSERT YOUR PROSEDURE.

00300 8000 STOP

00310 END

00320\$ EXECUTE

00330\$ FILE 08,A1S,100L

00340\$ LIMITS 6,50K,-4K,6000

00010##S,J N

-) コメント文で指定されているように、データ入力部分は利用者による編集が必要です。この部分では、データ書式変換プログラムの編集が可能（必要）です。

```

-LIST
00010##S,J N
00020$ JOB 6000810003$PASSWORD,B
00030$ LIMITS 25,,,6000
00040$ LOWLOAD
00050$ OPTION FORTRAN,RELMEM
00060$ FORTRAN LISTIN,BIN,NLNO,NFORM
00070CCC FORTRAN PROGRAM FOR DATA CONVERSION CCC
00080COMMENT : INSERT YOUR PROCEDURE.
00085 CALL ATTACH(01,'MEIBO;',1,1,ISTAT,)
00090C*** DECLARATION FOR VARIABLES ***
00100 CHARACTER NAMA*50,SKILL*20
00110 INTEGER CODENO
00120C*** END OF DECLARATION ***
00130COMMENT : INSERT YOUR PROCEDURE.
00140C*** UNIQUE ITEMS ***
00150 DO 7001 I01=1, 50
00160COMMENT : INSERT YOUR PROCEDURE.
00165 READ(01,100)NAME,CODENO
00166 100 FORMAT(A50,I5)
00170 IF(NAME.EQ.' ') GO TO 8000
00180 IREC=01
00190 WRITH(08) IREC,NAME,CODENO
00200C*** GROUP DNAME IS SKILLS ***
00210 DO 7002 I02=1, 0
00220COMMENT : INSERT YOUR PROSEDURE.
00225 READ(01,200) SKILL
00226 200 FORMAT(A20)
00230 IF(SKILL.EQ.' ') GO TO 7102
00240 IREC=02
00250 WRITE(08) IREC,SKILL
00260 7002 CONTINUE
00270 7102 CONTINUE
00280 7001 CONTINUE
00290COMMENT : INSERT YOUR PROSEDURE.
00295 CALL DETACH(01,ISTAT,)
00300 8000 STOP
00310 END
00320$ EXECUTE
00330$ FILE 08,A1S,100L
00340$ LIMITS 6,50K,-4K,6000

00295 CALL DETACH(01,ISTAT,)

-RESA * ←必ずRESAVEする。
DATA SAVED-CONVPROG

00295 CALL DETACH(01,ISTAT,)

-DONE

*** A DATA CONVERSION PROGRAM HAS BEEN GENERATED ***

!!! EDITTING CONVPROG JUST NOW !!!

```